An Efficient Linear Programming Based Method for the Influence Maximization Problem in Social Networks

Abstract—The influence maximization problem (IMP) aims to determine the most influential individuals within a social network. In this study, we first develop a binary integer program that approximates the original problem by Monte Carlo sampling. Next, to solve IMP efficiently, we propose a linear programming relaxation-based method with a provable worst-case bound that converges to the current state-of-the-art $1 - 1/e$ bound asymptotically. Experimental analysis indicate that the new method is superior to the state-of-the-art in terms of solution quality and this is one of the few studies that provides approximate optimal solutions for certain real life social networks.

Index Terms—Influence maximization, binary integer programming, sample average approximation, pipage method

1 INTRODUCTION

WITH the vast penetration of online social networks in our lives, the distribution of ideas, information or new products is carried out through social networks more and more frequently than ever. As a result, influential individuals are of great importance as they are highly demanded by companies for viral marketing campaigns [1]. The influence maximization problem (IMP) aims to determine the most influential individuals within a social network under interest. Aside from its viral marketing applications, the same concept is studied in network security, computer virus detection, epidemics analysis, infrastructure planning, habitat conservation, and wireless sensor networks [2], [3], [4], [5].

We define IMP as \{max $\sigma(S) : |S| = k; S \subset V$\} over a stochastic network. Here the set of nodes is represented as $V$ and the seed set $S$, which is a subset of $V$ with a fixed size $k$, corresponds to the influential nodes. Finally, the objective function $\sigma(S)$ is a measure of the expected number of nodes influenced in the social network when a cascade is triggered by the initial seed set $S$.

IMP has been attracting a great amount of interest in the last years and many researchers focus on different aspects of the problem [6]. The first model of IMP as a mathematical optimization problem is developed by Kempe et al. [7] (and its recent version [8]) where the problem is addressed as “Influence Maximization”. They show that the objective function $\sigma(S)$ is submodular under two popular diffusion models: Independent Cascade (IC) and Linear Threshold (LT). Consequently, a naive greedy algorithm together with Monte Carlo simulations to estimate $\sigma(S)$ guarantees a $(1 - 1/e)$ approximation to the optimal solution of IMP [9]. The requirement of calculation of the influence function $\sigma(S)$ many times in the Monte Carlo simulation steps results in scalability issues, especially for large networks. Upon this observation, many improvements have been proposed for the Greedy method such as Cost Effective Lazy Forward (CELF) method [2]. For large networks, even the performance of CELF is still unsatisfactory and different approaches are applied to solve IMP in terms of computational efficiency rather than the solution quality. Among the many, PMC [10], EasySIM [11] and IMM [12] are currently the best performers while keeping the approximate bound guarantee. In a recent work, Wang et al. [13] present a novel bottom-k sketch based RIS framework (BKRS), that significantly accelerates the seed set selection procedure. The author’s report that BKRS over-performs IMM. For a detailed comparison of the state-of-the-art, the survey by Li et al. [14] provides the expected and worst-case complexity levels of many algorithms.

While most researchers focus on scalability of IMP, some recent works focus on the solution quality. Sample Average Approximation (SAA) method is proposed in [3], [15] for obtaining approximate optimal solutions. Whereas, Wu and Küçükyavuz [16] develop a delayed constraint generation algorithm to find the optimal solution.

In this study, we focus on both the formulation and mathematical properties of IMP. Our first contribution is modeling the problem by using a simpler and much efficient discrete binary integer program than the one in our previous work [15], which is presented in Section 2.3. Our second contribution is a Linear Programming (LP) relaxation-based algorithm with a provable worst-case bound, which is a direct consequence of this new formulation. It provides a $(1 - (1 - 1/p)\epsilon)$ bound to the optimal solution, where $p$ is the size of the largest predecessor set whose any member can activate a given node in the network and $\epsilon$ is the error term due to sampling. We prove that this bound is as good as the current state-of-the-art $(1 - 1/e)$.

The paper is organized as follows: In Section 2, the preliminaries of the IMP, its mathematical formulation and how it is solved by SAA is given. Section 3 presents the LP-relaxation based algorithm and derivation of its worst case bound. In Section 4, we present the computational results obtained by using some real-life data. The last section gives the conclusion and future directions.
2 Basics of the Influence Maximization Problem

We define IMP over a weighted and directed network $\mathcal{N} = (\mathcal{V}, \mathcal{A}, \mathcal{W})$. In this network, $\mathcal{V}$ corresponds to the set of nodes that represent the members of the social network and its size is $|\mathcal{V}| = n$. The connections in the network such as being co-authors, followers, social media friends or similar relations are shown by arcs and the set of arcs is $\mathcal{A}$. Lastly, $\mathcal{W}$ is the set containing the weight information of these arcs. Each arc $(i, j) \in \mathcal{A}$ from node $i$ to $j$ has a weight of $w_{ij} \in \mathcal{W}$. It shows the probability of node $i$ influencing node $j$ whenever an action is triggered by node $i$ in the social network. The size of set $\mathcal{A}$ is $|\mathcal{A}| = m$. Given the set of initial influential (or seed set) $S$, which is the set of active nodes in the beginning of the cascading process, the expected number of activated or influenced nodes at the end of the process is calculated by the function $\sigma(S)$. IMP aims to identify the optimal seed set whose size is $k$ and when the diffusion process begins with the nodes in $S$, the influence function $\sigma(S)$ is maximized.

2.1 Independent Cascade Diffusion Model

It is assumed that the information propagates over the social network $\mathcal{N}$ obeying certain rules. To set these rules, some well-known diffusion models are used. In this study, independent cascade (IC) diffusion model is considered [17]. Regardless of the diffusion model, a node $i$ is called as active or influenced if it does the action of interest or accepts the idea that is shared to it. Whenever a node becomes active, it can not deactivate itself and stays as an active node throughout the diffusion process. Under the IC diffusion model a node $i$ that is activated in step $t$ has only a single chance to successfully influence any neighbour $j$ with a probability $p_{ij}$. These attempts are independent of each other and if node $i$ is unsuccessful, it has no chance to influence node $j$ any more. When a node $i$ activates its neighbour $j$ in step $t$, then node $j$ tries to activate all of its inactive neighbours in step $t + 1$. If there are more than one node trying to activate node $j$, they are sequenced arbitrarily. The process begins with an initial active seed set $S$ and runs until no more activations are possible [5], [7].

2.2 Evaluation of Influence

The influence function $\sigma(S)$ is computationally difficult for exact calculation. Chen et al. [5] prove that it is #P-hard to compute $\sigma(S)$ precisely. $\sigma(S)$ is equivalent to the expected number of nodes that can be accessed by the elements of the seed set $S$ of a corresponding random graph. Hu et al. [18] show that there are two possible methods for exact computation of $\sigma(S)$, similar to the computation of reliability or reachability both of which possess exponential-time complexity.

2.3 Mathematical Formulation of IMP

As mentioned in the introduction, IMP can be formulated as $\max \{\sigma(S) : |S| = k; S \subset V\}$. It is proven that IMP is NP-hard by showing a reduction to the Stochastic Set Covering problem [7]. Therefore, determining an optimal solution is very hard even for small networks.

This form of IMP is inconvenient for mathematical optimization. It can be reformulated as a binary integer program by enumerating all possible realizations (or scenarios) of the stochastic network. Each realization is matched by a subset of active and inactive arcs in the network. We call an arc $(i, j)$ as active if it exists in the network for a given realization. In the IC diffusion model the arc $(i, j)$ is either active with probability $p_{ij}$ or inactive with probability $(1 - p_{ij})$. Since the number of arcs in the network is finite, the number of realizations is large but finite. Let $\mathcal{R}$ be the set of all possible realizations and $r$ be the index of a given realization. Notice that $\mathcal{R}$ has an exponential size with $|\mathcal{R}| = 2^m$. Also, let $\mu_r$ be the probability of occurrence of realization $r$, which is computed by multiplying $m$ probabilities, i.e. $p_{ij}$ values for active arcs and $(1 - p_{ij})$ values for the inactive arcs.

In the proposed mathematical model, all nodes that can activate a given node $i$ should be identified a priori. For this purpose, we define $\mathcal{P}_{ir} \subseteq \mathcal{V}$, which is the set of all predecessor nodes of $i$ in the $r$-th realization. $\mathcal{P}_{ir}$ are constructed by running a breadth first search starting form node $i$ and going in the reverse direction by using only the active arcs in a given realization. Here, we use the probability weights $p_{ij}$ to determine if an arc is active or not. Notice that the higher the $p_{ij}$ value, the higher the chance of the corresponding arc $(i, j)$ to be an active one.

We define two sets of decision variables, one for representing the seed set and the second to capture all the activated nodes for each scenario. First, let $y_i$ be a $0-1$ vector of nodes in $\mathcal{V}$ and $y_i$ is one if node $i$ is selected as a seed and zero otherwise. Second, let $X(y)$ be the set of random variables corresponding to all nodes that are activated in the diffusion process given the seed set $y$. Each $x_{ir} \in X(y)$ is also a binary variable where $x_{ir}$ is one if it is activated in the diffusion process in sample $r$ or zero otherwise. The members of $X(y)$ with value one are the ones that correspond to the seed nodes plus the nodes which are activated in the later steps of the IC diffusion process. Notice that, the probability distribution ruling the activation of the arcs (therefore the nodes connected to them) is independent of $y$. Given these definitions the objective function $\sigma(S)$ is rephrased as:

$$\sigma(S) = \sigma(y) = \sum_{i \in \mathcal{V}} E[X(y)] = \sum_{i \in \mathcal{V}} \sum_{r \in \mathcal{R}} \mu_r x_{ir}.$$  

Also, the function $\sigma(y)$ has the same meaning with $\sigma(S)$ but its domain is the set $\{0, 1\}^m$. All the sets, parameters, decision variables and other relevant terminology are summarized in Table 1 for quick reference.

Following the above notations and definitions, the Influence Maximization Binary Integer Program (IMBIP) with independent cascade diffusion model is constructed as follows:

**IMBIP:**

$$\text{max } z = \sum_{r \in \mathcal{R}} \sum_{i \in \mathcal{V}} \mu_r x_{ir} \tag{1}$$

$$\text{s.t. } \sum_{i \in \mathcal{V}} y_i = k \tag{2}$$

$$x_{ir} \leq \sum_{j \in \mathcal{P}_{ir}} y_j \quad i \in \mathcal{V}, r \in \mathcal{R} \tag{3}$$

$$0 \leq x_{ir} \leq 1, y_i \in \{0, 1\} \quad i \in \mathcal{V}, r \in \mathcal{R} \tag{4}$$
TABLE 1: Sets, parameters, decision variables and related notations

<table>
<thead>
<tr>
<th>Sets</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V )</td>
<td>Set of nodes (</td>
</tr>
<tr>
<td>( A )</td>
<td>Set of arcs (</td>
</tr>
<tr>
<td>( W )</td>
<td>Set of arc weights</td>
</tr>
<tr>
<td>( S )</td>
<td>Seed set or initial set of influencers</td>
</tr>
<tr>
<td>( R )</td>
<td>Set of all possible scenarios or realizations</td>
</tr>
<tr>
<td>( P_i )</td>
<td>Set of all predecessor nodes who can activate node ( i ) in scenario ( r )</td>
</tr>
<tr>
<td>( R_{ij} )</td>
<td>Subset of scenarios obtained by Monte Carlo sampling</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k )</td>
<td>Seed set size</td>
</tr>
<tr>
<td>( p_{ij} )</td>
<td>Probability of node ( i ) to influence node ( j )</td>
</tr>
<tr>
<td>( \mu_r )</td>
<td>Probability of scenario ( r )</td>
</tr>
<tr>
<td>( R )</td>
<td>Number of possible scenarios or sample size</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decision Variables and Formulation Related Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y_{ir} )</td>
<td>Binary decision variable to represent seed nodes</td>
</tr>
<tr>
<td>( z )</td>
<td>Objective function value for IMBIP</td>
</tr>
<tr>
<td>( y^* )</td>
<td>Optimal seed set (or optimal solution) of IMBIP</td>
</tr>
<tr>
<td>( z^* )</td>
<td>Optimal objective function value of IMBIP</td>
</tr>
<tr>
<td>( z^*_R )</td>
<td>Optimal seed set of IMBIP-S</td>
</tr>
</tbody>
</table>

Sample Average Approximation Related Notation

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of batches (integer programs) to be solved</td>
<td>( M )</td>
</tr>
<tr>
<td>Sample sizes used in step-1 ( 1, 2, 3 ) of SAA algorithm</td>
<td>( R_1 )</td>
</tr>
<tr>
<td>Objective function value of the ( i )-th IMBIP-S in step-1</td>
<td>( z_{iR_1} )</td>
</tr>
<tr>
<td>Objective function value of the ( r )-th IMBIP-S in step-1</td>
<td>( y_{ir} )</td>
</tr>
<tr>
<td>SAO obj. funct. value (upper bound estimate)</td>
<td>( z_{iR_1} )</td>
</tr>
<tr>
<td>Point estimate of ( z_{iR_1} ) with a larger sample size in step-2</td>
<td>( z_{iR_2} )</td>
</tr>
<tr>
<td>Best seed set yielding the highest ( z_{iR_2} ) value</td>
<td>( y_{iR_2} )</td>
</tr>
<tr>
<td>Final point estimate for the seed set ( y_{iR_2} ) (lower bound estimate)</td>
<td>( z^*_R )</td>
</tr>
</tbody>
</table>

IMBIP aims to maximize the objective function (1) which calculates the expected number of activated nodes. The constraint (2) limits the size of the seed set to \( k \), whereas, the constraints (3) capture the diffusion process. A node \( i \) in realization \( r \) is active only when it is in the seed set or it is connected to any member of the seed set through active arcs, which is captured by \( P_i \). The formulation is completed with binary restrictions on \( y \) and non-negativity and unit bounds on \( x \) (4). Notice that, even though \( x_{ir} \) are initially defined as binary variables, the objective function and constraints (3), (4) force them to be either 0 or 1, therefore we can relax the binary requirements. When solved optimally, IMBIP determines the optimal seed set \( y^* \) which attains the maximal influence value of \( z^* \).

Sheldon et al. [4] developed a similar formulation, which is valid for only acyclic graphs. However, our formulation is superior as it can be used successfully for networks with cycles. For cyclic networks, when no precautions are taken in the formulation, a solution may contain some nodes becoming active without any of them being connected to any of the initial active nodes. Such solutions are unacceptable and it can be avoided by determining the predecessor set as we have done, or by using a less efficient formulation that requires an additional time index on the decision variables [15].

### 2.4 Solving IMP by Sample Average Approximation

Solving IMP (and also IMBIP) optimally is not practical as a consequence of the excessive number of possible realizations |R|. Therefore, approximating \( \sigma(S) \) using various sampling methods is a viable strategy. Among all, Monte Carlo sampling is used frequently to obtain a representative sub-network. For this purpose, a biased coin is flipped for each arc \((i,j)\) with a success probability of \( p_{ij} \). Thus, a sub-network \( \mathcal{N}_r = (\mathcal{V}, \mathcal{A}_r) \) is constructed for each realization \( r \), that contains only the active arcs according to the sampling outcome. Let \( \mathcal{R}_s \subset \mathcal{R} \) denote the set of the realizations obtained by Monte Carlo sampling and let \( \sigma_r(y) \) be the expected number of nodes reachable in sub-network \( \mathcal{N}_r \) when the diffusion process is triggered by the seed vector \( y \).

By sampling we generate a total of \(|\mathcal{R}_s| = R\) realizations and we define \( z_R(y) = \frac{1}{R} \sum_{r \in \mathcal{R}_s} \sigma_r(y) \), which is a Monte Carlo Sample Average Approximation (SAA) of \( \sigma(y) = \sigma(S) \) [3], [15]. Observe that \( z_R(y) \) is an unbiased estimate of the original objective function \( \sigma(S) \).

Then the SAA version of the optimization problem becomes:

#### IMBIP-S:

\[
\begin{align*}
\max \quad & z_R = \frac{1}{R} \sum_{r \in \mathcal{R}_s} \sum_{i \in \mathcal{V}} x_{ir} \\
\text{s.t.} \quad & \sum_{i \in \mathcal{V}} y_{ir} = k \\
& x_{ir} \leq \sum_{j \in \mathcal{P}_i} y_{jr} \quad i \in \mathcal{V}, r \in \mathcal{R}_s \\
& 0 \leq x_{ir} \leq 1, y_{ir} \in \{0, 1\} \quad i \in \mathcal{V}, r \in \mathcal{R}_s
\end{align*}
\]

Note that the major difference in this formulation is the new objective function and the much smaller realization set \( \mathcal{R}_s \). Let \( y^*_R \) denote the optimal solution (seed set) to IMBIP-S. The fact that the number of all possible \( k \) combinations of \( y \) is finite (but obviously very large) guarantees that \( \lim_{R \to \infty} z^*_R = \sigma(y^*_R) \) with probability one and similarly the limit points \( \{y^*_R\} \) is optimal for IMBIP-S with probability one. Norkin et al. [19] proved that \( E[z^*_R] \geq \sigma(y^*_R) \) due to expectation relaxation. In other words, the optimal objective function value of IMBIP-S is a positively biased estimator of the true optimum, even though the SAA function itself is not. Thus for a more accurate objective function estimate, a function evaluation, which is simply the point estimation of \( z^*_R \) is necessary. For this purpose, a much larger sample size \( R' \gg R \) is used to compute \( z_{R'} = \frac{1}{R'} \sum_{r \in \mathcal{R}_s'} \sum_{i \in \mathcal{V}} x_{ir} \) for the best SAA solution. Notice that computation of \( z_{R'} \) only requires the simulation of the diffusion for a fixed set of seed nodes, therefore it is very fast since it does not include any combinatorial optimization. To frame it up, the SAA procedure computes both an approximate upper bound \( z^*_R \) and a feasible (hopefully optimal) objective function estimate \( z^*_R \) for the true optimum \( \sigma(y^*_R) \). One can refer to [15] for the detailed implementation of SAA method that is used in this work, which is slightly different from [3]. However, for completeness we provide steps of the SAA method in Algorithm 1.

SAA is applied as a three step procedure. In the first step, \( M \) independent integer programs are solved which are constructed by \( M \) iid batches of \( R_1 \) samples. Both the solutions and the objective function values are recorded. The average of these \( M \) solutions, \( \bar{z}_{R_1} \) is recorded as an upper bound estimate to the original objective function. In the second step, the optimal solutions of each integer program of first step are compared with more accurate estimations. For this purpose, we algorithmically evaluate the objective function values, \( z^*_{R_1}(y^*_R) \), by using a much...
Algorithm 1 SAA Algorithm

1: Use $M$ batches of samples of sample size $R_1$ for solving $M$ integer programs to obtain the solutions $y^*_R$ and objective function values $z^*_R, i = 1, ..., M$. Let $\hat{z}_{R_1} = \frac{1}{M} \sum_{i=1}^{M} z^*_R$, and record it as an upper bound estimate to $\sigma(S)$.
2: Compute the influence values (point estimates) $z^*_R(y^*_R)$ algorithmically, for the solutions $y^*_R, i = 1, ..., M$ with a larger sample size $R_2$. Identify the solution with the best objective function value i.e., $y^* = \arg\max_x \{z^*_R(y^*_R)\}$ and record it as the optimal seed set estimate.
3: Compute $z^*_R(y^*)$ algorithmically with a large sample size of $R_3 \geq R_2$ and record it as a lower bound estimate to $\sigma(S)$.

larger sample size $R_2$. The solution $y^*$ yielding the highest objective function in step two is assumed to be the optimal solution (seed set) of the SAA method. In the final step of the SAA method, the objective function of the best solution, $y^*$, is algorithmically re-computed with a new sample of size $R_3$, which is generally taken larger than or close to $R_2$. Finally, $z^*_R$ and $z^*_R$ are reported as the estimates of the optimal seed set and optimal objective function value of the original problem. Notice that, since $y^*$ is a feasible solution, $z^*_R$ is a lower bound (estimate) to the optimal problem. Therefore we have the approximate relation on the optimality gap: $\hat{z}_{R_1} \geq \sigma(S) \geq z^*_R$.

2.5 Quality of Approximations

A critical question arises about how to determine the sample size $R$ (consequently $R_1, R_2, R_3$ and $M$) to obtain (i) a good (optimal) seed set and (ii) a good estimate of the original objective function $\sigma$. For the first case, we refer to the approximation results in Kleywegt et al. [20]. Let $y^*_R$ and $\hat{y}^*_R$ be the set of all $\rho$-optimal solutions to IMBIP and IMBIP-S, respectively. By using the theory of large deviations (LD), they show that the probability of the event of having the optimal SAA solution ($y^*_R$) as the true optimum solution ($y^*_R$) converges to one exponentially as $R \rightarrow \infty$. By also benefiting from Cramér’s LD theorem, they show that with a fixed significance level $\alpha \in (0, 1)$, the probability $P(\hat{y}^*_R < y^*_R) \geq 1 - \alpha$ i.e., any $\rho$-optimal solution of the SAA problem is an $\epsilon$-optimal solution to the original problem with probability at least $1 - \alpha$, whenever the sample size, $R \geq \frac{3\sigma^2_{\max}}{(\epsilon - \rho)^2} \log \left( \frac{|X|}{\alpha} \right)$ (9) times, then the average number of activated nodes over these runs is a $(1.\pm \epsilon)$ approximation to $\sigma(y)$, with probability at least $1 - \delta$ [8].

As a summary, our first contribution is the new formulation of IMBIP and thus IMBIP-S. It is a simpler formulation compared to the one in our previous work [15], where the decision variables have an additional time index $t$ showing the actual step when a node is activated in the diffusion process. A breadth-first search procedure is run to find at which step a node is activated for each of its predecessors that can access to node $i$. Then the maximum of these values is recorded as $T^\rho_{\max}$ for each node-sample couple. In that formulation, the predecessor set is not kept (although implicitly identified), but just the one-hop neighbours are kept as the neighbourhood set $N^t_i$. to construct the constraints. In the new formulation, we again apply the breadth-first search procedure and this time record the predecessor set $\mathcal{P}_i$ for each node-sample couple. This process is similar to in the length of the problem input (number of nodes and sample size), the variance $\sigma^2_{\max}$ grows polynomially in the length of the problem input. Also the complexity of finding a $\rho$-optimal solution for the SAA problem grows polynomially in the length of the problem input. Then a solution can be generated in time that grows polynomially in the length of the problem input such that, with probability at least $1 - \alpha$, the solution is $\epsilon$-optimal to the original problem.

The computational complexity of solving the IMBIP-S as a stand-alone integer program grows exponentially in the number of decision variables and thus the sample size. Therefore, instead of solving a single IMBIP-S with a large sample size $R$, we prefer to use a smaller sample size $R_1$ and solve several IMBIP-S problems with i.i.d. samples. This approach is generally recommended in the SAA literature for better computational performance [20], [21], [22]. Each SAA problem can be thought as a Bernoulli trial with probability of success $p = p(R_1)$. Here success means that the optimal seed set identified by the SAA problem is the actual optimal seed set of the original problem. As mentioned above, this probability $p$ tends to one as $R \rightarrow \infty$ exponentially fast. The probability of producing an optimal seed set of the true problem at least once in $M$ trials is $1 - (1 - p)^M$ and this probability again tends to one exponentially as $M \rightarrow \infty$, given that $p > 0$. Therefore, instead of solving a single SAA problem with sample size $MR_1$, solving $M$ separate SAA problems with sample size $R_1$ is preferable from the computational effort needed to solve integer programs. Given this conjecture, there is no guarantee that $p > 0$ for a given sample size $R$ (or $R_1$). Particularly, for a finite $R$ the probability $p$ can be very small or even zero. Hence, even though this practical approach has empirical success, it does not contribute to the theoretical bound on the sample size $R$.

The choice of sample size at the second and third steps of the SAA method is similar to other IMP approaches. Classical Chernoff-Hoeffding bounds are valid as we algorithmically simulate the diffusion process for a fixed seed set $y$. Therefore, if the diffusion process is repeated for a given seed set $y$ independently at least

$$R \geq \frac{n^2 \log \left( \frac{1}{\delta} \right)}{2}$$

(10)

where

$$P(\hat{y}^*_R < y^*_R) \geq 1 - \alpha$$
the reverse reachable set notion presented in [23] and the following studies. As a consequence, the time index can be dropped because it is not needed to construct the constraints any more. This approach significantly reduces the number of variables and constraints, thus larger instances of IMBIP-S can be solved even as stand-alone integer programs with sampling. More importantly, the new form of IMBIP-S has a special structure that facilitates the development of an LP-based solution technique, which is discussed in the next section.

3 An LP Relaxation Based \( C \)-Approximation Algorithm

Scalability is a critical issue in influence maximization. Many real-life social networks contain millions of nodes and up to billions of edges. As a result, most of the well-known studies in the literature are the ones that propose fast and scalable methods with relatively less emphasis on optimality. Nevertheless, these algorithms try to keep their worst case bound to a reasonable level, mostly to \( 1 - 1/\epsilon \). However, in this work, our objective is to find \( \epsilon \)-optimal solutions and improve our scalability with minimum sacrifice from optimality. To achieve this, we benefit from the LP-relaxation of IMBIP-S.

In IMBIP-S, the seed set variables \( y_i \) dictate the optimal values for \( x_{ir} \) together with the unit bounds. Therefore, maximizing \( z \) is equivalent to maximizing the non-linear function \( F(y) = \sum_{r \in R} \sum_{i \in V} \left( 1 - \prod_{j \in P_{ir}} (1 - y_j) \right) \) over all binary vectors \( y \) satisfying (6). Similarly, the objective function \( z \) can be replaced by the function \( L(y) = \sum_{r \in R} \sum_{i \in V} \min\{1, \sum_{j \in P_{ir}} y_j\} \), which is favorable for an LP-relaxation based optimization scheme.

Ageev and Sviridenko [24] have shown that for the maximum \( k \)-coverage problem the following two conditions hold:

(i) The function \( \psi(\epsilon, y, s, t) = F(y_1, ..., y_s + \epsilon, ..., y_t - \epsilon, ..., y_n) \) is convex with respect to \( \epsilon \in [\min\{y_s, 1 - y_t\}, \min\{1 - y_s, y_t\}] \) for each pair of indices \( s \) and \( t \) and each \( y \in [0, 1]^n \).

(ii) There exists \( C > 0 \) such that \( F(y) \geq CL(y) \) for each \( y \in [0, 1]^n \).

Observe that IMBIP-S is a special case of the SAA version of stochastic maximum \( k \)-coverage problem. The objective function aims to find a ground set with size \( \epsilon \) that maximizes the average coverage for the given \( R \) instances of the network. The constraint (6) restricts the number of selected variables from the ground set to \( k \) and lastly the constraints (7) are equivalent coverage constraints. Different than the classical maximum \( k \)-coverage problem, in IMBIP-S, the \( i \)-th constraint of (7) contains both \( x_{ir} \) and \( y_i \) at the same time for any \( r \). So if a node is not connected to any other node, then \( P_{ir} \) contains only \( i \), otherwise it contains \( i \) plus all the nodes that can access \( i \) in sample \( r \). We will first show that these two conditions are also valid for IMBIP-S.

Proposition 1. For IMBIP-S, the function \( \psi(\epsilon, y, s, t) = F(y_1, ..., y_s + \epsilon, ..., y_t - \epsilon, ..., y_n) \) is convex with respect to \( \epsilon \in [\min\{y_s, 1 - y_t\}, \min\{1 - y_s, y_t\}] \) for each pair of indices \( s \) and \( t \) and each \( y \in [0, 1]^n \).

Proof. \( \psi(\epsilon, y, s, t) \) will contain terms like \( (1 - (1 - y_s - \epsilon)(1 - y_t + \epsilon) \prod_{j \in P_{ir}/(s,t)} (1 - y_j) \) or will have \( y_s \) and \( y_t \) in separate products according to the network structure and the sampling outcome. Therefore the main coefficient of \( \epsilon \) will always be either quadratic (if they are in the same product) or linear (if they are in separate products). Thus \( \psi(\epsilon, y, s, t) \) is always convex for each pair of indices \( s \) and \( t \) and each \( y \in [0, 1]^n \).

Proposition 2. There exists \( C > 0 \) such that \( F(y) \geq CL(y) \) for each \( y \in [0, 1]^n \), where \( C = (1 - (1 - 1/p)^p) \), and \( p = \max\{|P_{ir}| : i \in V, r \in R_s\} \).

Proof. Let \( g(z) = 1 - (1 - z/p)^p \), where \( z = \min\{1, \sum_{j \in P_{ir}} y_j\} \).

By using the arithmetic-geometric mean inequality we can write:

\[
1 - \prod_{j \in P_{ir}} (1 - y_j) \geq 1 - (1 - z/p)^p
\]

Observe that \( g(z) \) is concave between \([0, 1]\) and \( g(0) = 0, g(1) = 1 - (1 - 1/p)^p \), so \( g(z) \geq (1 - (1 - 1/p)^p)z \).

Therefore,

\[
1 - \prod_{j \in P_{ir}} (1 - y_j) \geq g(z) \geq (1 - (1 - 1/p)^p) \left( \min\{1, \sum_{j \in P_{ir}} y_j\} \right)
\]

which completes the proof.

Now, consider the LP-relaxation of IMBIP-S and its optimal solution \( y_{LP} \). If \( y_{LP} \) is integral, then it is the optimal solution to its original integer version as well. Otherwise, there exists at least 2 fractional variables \( y_s \) and \( y_t \) due to (6). Of these two fractional variables, at least one of them can be made binary by adding or subtracting one of the two possible values of \( \epsilon \), which is either \( \epsilon = \min(1 - y_s, y_t) \) or \( \epsilon = -\min(y_s, 1 - y_t) \). Since we are maximizing, the \( \epsilon \) yielding the higher function value is preferred. This fractional-to-integer conversion of a fractional variable is called ‘pipage’. Then, the new feasible solution \( y_{LP} = \{y_1, ..., y_s + \epsilon, ..., y_t - \epsilon, ..., y_n\} \) has less non-integer components. After repeating these ‘pipage’ steps at most \( n - 1 \) times, a binary feasible solution \( y_{LP} \) is obtained.

Theorem 1. The pipage method is a \( C \)-approximation algorithm for IMBIP-S, where \( C = (1 - (1 - 1/p)^p) \), and \( p = \max\{|P_{ir}| : i \in V, r \in R_s\} \).

Proof. First, we will show that the worst case bound is asymptotically close to \( C = (1 - (1 - 1/p)^p) \). Then we will explain how pipage method guarantees a feasible integer solution preserving the integrality gap.

Set \( n = pk \). Let \( P_i \) be the collection of all subsets of \( \{1, 2, ..., n\} \) \{-\{i\}\} with cardinality \( p - 1 \). We want to compute the lowest value for \( z_{LP}(IP)/z_{LP}(LP) \) for IMBIP-S. Remember that the constraints (7) are constructed as \( x_{ir} \leq y_i + \sum_{j \in P_{ir}/(i)} y_j \) with a total of \( p \) terms on the RHS, because the seed set variable \( y_i \) is always included in the constraint of \( x_{ir} \) for every sample. Then by symmetry any binary vector \( y \) with \( k \) units will provide a solution \( z_{LP}(IP) = k^{(n-1)} + (n - k) \left( \binom{n-1}{p-1} - (n-k-1) \right) \). In this summation, the left term counts all \( x_{ir} \) having the same index with the elements of the seed set. Whereas, the right
term is the expected influence of the remaining nodes that are activated later by the diffusion process, which occurs only when the predecessor set of $z_{IP}$ contains at least one of the $k$ seed nodes for the corresponding sample. For the LP relaxation, the vector with all $y_i = 1/p$ provides the optimal LP solution with $z_{IP}^* = n(n-1)/p$, which means all nodes are activated in all samples. Now we can compute the ratio $z_{IP}^*(IP)/z_{IP}^*(LP)$ to provide an upper bound on the optimality gap $C$. 

$$\frac{z_{IP}^*(IP)}{z_{IP}^*(LP)} = \frac{k(n-1) + (n-k)(n-k-1)}{n(n-1)} = \frac{n(n-1) - n(n-k-1)}{n(n-1)}$$

Multiplying both the nominator and denominator with $1/p$ we get:

$$\frac{z_{IP}^*(IP)}{z_{IP}^*(LP)} = \frac{(n-1) - (n-k)}{p}$$

This is the same bound obtained in [24] so the proof follows:

$$\frac{z_{IP}^*(IP)}{z_{IP}^*(LP)} = 1 - \frac{n-k}{p(n-k-p)} \leq 1 - \frac{n-k}{n} \leq 1 - \frac{1}{p} \leq 1 - \frac{1}{p} \leq 1 - \frac{1}{p}$$

and the last term tends to $(1 - (1 - 1/p)^p)$ when $p$ is fixed and $n \to \infty$.

Assume that the LP-relaxation of IMBIP-S provides a fractional solution $y_{IP}^*$. After applying a pipage step to obtain the new solution $y_{LP}^*$ with less fractional variables, $F(y_{LP}^*) \geq F(y_{IP}^*)$ will always hold due to convexity of $\psi(x, y, s, t)$ for $\epsilon \in [-\min\{ys, 1 - ys\}, \min\{1 - ys, ys\}]$. When all the pipage steps are completed with no fractional variables left, a binary feasible solution $y_{IP}^*$ with $F(y_{IP}^*) \geq C\min\{1, 1 - 1/p\}^p = CF(y^*)$ is obtained, where $y^*$ is the actual optimal integer solution to IMBIP-S.

The pipage method has a polynomial time complexity. It involves, first solving an LP relaxation of IMBIP-S, plus the additional pipage steps, which are simple function evaluations. Also its worst case bound is as good as the greedy method. The greedy method solves IMBIP-S within a factor of $\left(1 - (1 - 1/k)^k\right)$ [9]. For the asymptotic case our bound $(1 - (1 - 1/p)^p)$ behaves similar to $\left(1 - (1 - 1/k)^k\right)$, but for fixed $p$, the performance guarantee of the pipage method beats that of the greedy method. For instance, when the underlying network is bipartite, pipage method’s bound strengthens to $3/4$, whereas the greedy method has still the $1 - 1/e$ bound [24].

We can easily integrate the pipage method to our solution approach. In the first step of Algorithm 1, instead of solving IMBIP-S as binary integer program, we solve its LP relaxation. If the solution is integer the algorithm proceeds just as it is. However, if the solution is fractional we obtain an integer solution by using the pipage method. So at the end of step one, we have $M$ objective function values of the relaxed problems and $M$ integer seed sets. These objective function values are used to compute the estimated upper bound $(z_{IP^*})_S \geq (z_{IP^*})_P$ and the corresponding seed sets are used in step-2 of the algorithm where the best seed set is identified by simulation as before. Basically, we lose from the tightness of our upper and lower bound estimates, but in return we guarantee a method with polynomial time complexity and potential for scalability. Overall the SAA method with pipage provides a $(1 - (1 - 1/p)^p) - \epsilon$ approximation to IMBIP, where the $\epsilon$ is the approximation error that is carried to IMBIP-S from IMBIP due to sampling.

## 4 Experimental Results

In this section we present our experimental results. First, we compare the performance of our new formulation provided in Section 2.3 with our old formulation in [15]. Next, we analyze the performance of our LP-based approach and compare both its solution quality and computational performance with the integer programming (IP) version and two other popular methods from the literature.

### 4.1 Experiment Setup and Data Sets

Three data sets are used in our experiments which are available in SNAP (https://snap.stanford.edu/data) and a brief summary of the networks are given in Table 2. The first one is High Energy Physics (HEP) section of arXiv data, which is also used in the experiments of [5], [7], [8], [15]. In this data set each node corresponds to an author and each arc is a co-authorship relation among authors. The co-authorship network displays a scale-free network characteristic, where a small number of nodes (authors) have high number of connections and most of the nodes have low number of connections [15], [25].

The second data set is ego-Facebook, which is a small world type network containing social circles (friend lists) from Facebook with 4, 039 nodes and 88, 234 arcs. The last data set is an internet peer-to-peer network example, P2P-Gnutella04, with 10, 876 nodes and 39, 994 arcs. A sequence of snapshots of the Gnutella peer-to-peer file sharing network from August 2002 are collected where nodes represent hosts in the Gnutella network topology and edges represent connections between the Gnutella hosts. This data set is also used in [16].

TABLE 2: The Summary of Data Sets

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Category</th>
<th>Nodes</th>
<th>Edges</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEP arXiv</td>
<td>scale-free</td>
<td>37,153</td>
<td>231,568</td>
<td>Undirected</td>
</tr>
<tr>
<td>ego-Facebook</td>
<td>small world</td>
<td>4,039</td>
<td>88,234</td>
<td>Undirected</td>
</tr>
<tr>
<td>P2P-Gnutella04</td>
<td>peer-to-peer</td>
<td>10,876</td>
<td>39,994</td>
<td>Directed</td>
</tr>
</tbody>
</table>

For the arXiv data set, we create 6 different sized networks with the number of arcs $m = \{1000, 2000, 5000, 10000, 20000, 50000\}$ that are selected randomly from the master dataset and 30 different seed set sizes, $k = 1, ..., 30$, meaning a total of 180 different test scenarios. For the ego-Facebook and P2P-Gnutella04, the complete data sets are considered and only seed set size is varied from $k = 1$ to $k = 30$. For the undirected networks, we assume the first node to be the tail and the second node to be the head. Placing two edges in both directions is a more common way of handling
undirected networks, but since the networks are already large for the integer programs, we preferred such a setting for scalability. Since we are not interested in determining the actual best influential nodes of the given networks, but rather testing our approaches on random networks, this choice does not hurt our experimental analysis.

4.2 Methods Applied to Obtain Seed Sets
A total of four methods (IP, Pipage, CELF and IMM) are compared by testing their performance on the mentioned three data sets. To have a fair and accurate comparison of solution quality, we determine the best possible seed sets by each method with their default settings. Then the expected influence of these seed sets are computed algorithmically by simulating the diffusion process on the same batch of sampled networks with a large sample size.

The first method, IP, is the classical implementation of SAA as displayed in algorithm 1, where we solve integer programs (IMBIP-S) at the first step. In method Pipage, we again use algorithm 1, but in the first step we solve the LP relaxation of IMBIP-S and apply the pipage method as described in Section 3 to obtain a feasible seed set if the solution is fractional. In both IP and Pipage methods, the Monte Carlo SAA sample size $R_1 = 100$ and the number of SAA replications $M = 25$. So basically, $M = 25$ instances of the BIP or LP are constructed with a sample size $R_1 = 100$ and then solved by using a commercial solver. For the point estimates (objective function evaluations) of the second step of the SAA procedure, the extended sample size is taken as $R_2 = 10,000$ [7], [15]. Notice that, since we need to identify the best seed set, we don’t need to execute the step-3 of algorithm 1. The choice of $M \in [20 - 30]$ is recommended in [21] to induce Central Limit Theorem for providing confidence intervals on the upper bound estimates. The choice of $R_1 = 100$ is an empirical preference, where the variance of the estimates start to stabilize around this level. We provide an extended analysis showing the quality of estimates with respect to varying values of $R_1$.

Our third method is the CELF implementation of the greedy method which is one of the fastest versions of it [2]. The sample size used in the greedy method is also $R = 10,000$ [7]. Last method is IMM - Influence Maximization with Martingales. It is one of the fastest algorithms to solve IMP with $1 - 1/e$ worst-case objective value guarantee. The IMM method is applied with its default parameter values where we set $\epsilon = 0.1$ [12].

The experiments are run on a server with two 64-bit, 3.00-GHz Xeon processors and 32GB Ram memory and the operating system is Windows 2008 R2 Server. GUROBI 7.5 is used for solving the BIPs and LPs [26] with one hour time limit. If the problems are not solved within the time limit, for the upper bound we take the best bound given by Gurobi and for the seed set we take the best feasible solution in hand. Saying this, none of the IP’s or LP’s exceeded the time limit in our experiments.

4.3 Comparison of IMBIP Formulations
Our first analysis is on the efficiency of our new BIP formulation over our old formulation [15]. The tests are carried out on the arXiv data set with $m = \{1000, 2000, 5000, 10,000, 20,000, 50,000\}$ and $k = 1, \ldots, 30$. Algorithm 1 is applied with the aforementioned parameters. The running times are recorded and averaged over all $k$ values and displayed in Figure 1, i.e. the values displayed are averages of 30 results. Observe that the new formulation is 2 to 30 times faster than the old one, which is parallel to our expectations as the new formulation has much fewer number of variables and constraints. With this lighter formulation we can solve network instances with $m = 100,000$ from arXiv data set, which is twice as large as our previous largest instance.

![Fig. 1: Running Time Comparison of BIP Formulations](image-url)

4.4 Solution Quality of the Seed Sets
In this sub-section, we make a comparison on the solution quality of the proposed methods. To have a fair comparison, the expected influence of each seed set obtained by each method are computed over the same sampled network with a large sample size $R = 10,000$. With this setting, if two methods have found the same seed sets, they will definitely produce the same expected influence, so that there will not be any deviation in objective function values due to sampling. On the contrary, when the seeds sets are different, we assume that the one yielding a larger influence is a better one with high probability and for very close values, our decision may be wrong but from a computational perspective the difference is assumed to be statistically insignificant.

For arXiv data set, a representative sample of 24 out of 180 test scenarios is listed in Table 3. The first column shows the number of nodes and arcs of the sampled network and the seed set size in order. The second column shows the values of the point estimate of the best SAA solution ($z^*_R(y^*_S)$), which is tagged as $IP$. This value is actually the best estimate for the true optimum objective function value $\sigma(S)$ in our setting. The last three columns display the percent gap between point estimates of IP and the point estimates of the pipage, greedy and IMM methods. The gap is calculated with the formula $\frac{z^*_R(y^*_S) - a}{b} \times 100$, where $a$ is $z^*_R(y^*_S)$ and $b$ is the point estimate value for the corresponding method. One can see that both the pipage and greedy method are performing quite well with very low gaps (0.13% and 0.30% on the average) and they find the same best solution in most of the cases. The gap for IMM is larger with 2.11% but again it is far from its theoretical worst case bound.
Table 4 displays the results for the Facebook data set. Again, Pipage and CELF are performing close to the best integer results with an average gap of 0.48% and 0.84%, respectively. IMM’s performance is slightly worse than the others with an average of 2.98% gap. Observe that, Facebook network is denser than the other two networks, which increases the complexity of finding good seed sets and thus IP’s performance superiority increases. Nevertheless, the differences between IP, Pipage and CELF are not statistically significant.

The results on our last data set Gnutella are somewhat controversial, which are presented in Table 5. Here IMM is the best performer, followed by Pipage, IP and CELF in order. This network is very sparse, hence the amount of sampling plays a significant role. As IMM method creates lots of samples compared to the other methods, it can identify the most influential seeds better than the others. To illustrate it better, consider the instance of Gnutella network where \( n = 10879, m = 39993, k = 30 \). For this instance, IMM creates 247 reverse reachable set samples per node on the average, whereas this value is 100 in both IP and Pipage. In denser networks, the effect of under sampling is not as critical as in this sparse network.

### 4.5 Effects of LP Relaxation and Pipage Method on Upper and Lower Bound Estimates

As mentioned earlier, solving the LP relaxation of IMBIP-S in the SAA algorithm results in higher upper bound estimates. Similarly, rounding the fractional solutions by pipage method to get integral solutions may weaken the lower bound estimates. For the worst case, the degradation can be up to \( 1 - 1/e \) of the optimal solution. Table 6 provides useful insights to see the amount of degradation of the upper bounds. In Table 6, the second column displays the average optimality gap between the optimal solution and the LP relaxation solution, and the third column displays the average optimistic gap between the LP relaxation solution and the pipage method solution.
number of times the SAA upper bounds ($\hat{z}_{R_k}$) obtained by IP and LP Relaxation are not equal for $k = 1, \ldots, 30$ cases. Remember that in each case, we solve $M = 25$ problems and average it to obtain $\hat{z}_{R_k}$. So the SAA objective function values of the two method will be equal only when all the 25 individual function values are exactly the same for both methods (i.e. yielding the same seed sets unless there is a multiple optima). Notice that, when the network gets larger or denser, the number of times we see a deviation increases. For smaller or sparse networks the bounds are identical most of the time, which means LP relaxation provides the same integer solution (or even fractional the same objective function value meaning that there is a multiple optima). The third column provides the average integrality gap between IP and LP objective function values. It is computed for only the cases with non-zero integrality gap. So for instance, in arXiv(m=50K) case, 27 non-zero gap values are summed and divided by 27 to obtain 0.039%. The average gaps are very small and the largest difference occurs for the Facebook data set which is still smaller than 0.2%. Therefore we can conclude the LP relaxation of IMBIP-S is very strong with a very little integrality gap for the social networks under consideration. Theoretically LP relaxation can be as large as $e/(e-1)$ times the IP value, which is roughly 57%, but such a case is never observed. The last column shows the average number of pipage steps when a fractional solution is found. Again as the network gets denser and more connected the LP relaxation yields higher number of fractional $y$ values and increases the number of pipage rounding steps.

One final insight is provided in Figure 2. It shows the relationship between average integrality gap and the seed set size. These are the average gap values computed over 8 data sets (arXiv, Facebook and Gnutella) for each $k$. For $k = 1$ the gap is zero and as the seed size increases, the gap gradually increases but even for $k = 30$ it is still lower than 0.1%.

The comparison of lower bound estimates (thus the quality of seeds obtained by pipage method) is already available in Tables 3-5. Observe that, the gap between IP and Pipage methods’ lower bound estimates are always very low. Out of a total 84 results listed, the highest difference observed is 1.47% (in the Facebook network). Therefore, we can assess that the strength of LP relaxation of IMBIP-S is reflected to the lower bound estimates as well.

4.6 Comparison of Running Times

We compare the computational performance of four methods and they are displayed in Figure 3. The values are the averages over $k = 1, \ldots, 30$ seed set scenarios. As expected IMM is very fast compared to the remaining three methods. In the slowest case it requires about 14 seconds to find the seed set. The running time of IP and Pipage are very close to each other and Pipage is slightly faster. Except a few cases greedy method is the slowest one and the gap increases as the data sets get larger and denser.

Figure 5 provides an interesting insight. It is obtained by averaging the computation times over 8 data sets for each $k$. It shows that our SAA based approach is quite insensitive to increasing seed set size $k$, however, greedy method’s running time is almost linearly growing with respect to $k$. When $k$ is small CELF is faster than SAA, but after a certain value of $k$ (around $k = 5$ for large problems and $k = 15$ for small problems) our method becomes faster. Thus, especially for the instances with large seed set sizes our method is computationally favorable. Overall the contribution of our LP-based approach’s computational efficiency is not very significant. This is mainly due to the fact that Gurobi Solver is solving both IP and LP versions of the given instances almost with identical running times. This phenomenon must be further investigated to see if there are some special structures in the problem that are exploitable using LP theory for better performance.

Lastly, we analyzed the effect of sample size $(R_1)$ on the solution quality and running time. Figure 4 summarizes our findings. Here we tested different values of $R_1$ varying between 15 to 225 for fixed $k = 20$ on four different data sets (arXiv 10K, arXiv 50K, Facebook, Gnutella). The left $y$-axis is displaying the standard deviation of the $M = 25$ upper bound values ($z_{R_1}$) from their mean ($\hat{z}_{R_1}$). The lines show the values for each corresponding data set and observe that the standard deviation begins to stabilize around $R_1 = 100$. The running times are also significantly affected from increasing sample size, which are displayed as vertical bars (values displayed on the right $y$-axis with logarithmic running time scale). Observe that, the running times increase considerably after $R_1 = 50$. Therefore, a choice of $50 \leq R_1 \leq 100$ is reasonable. To summarize, the choice of sample size critical to determine the trade-off between running time performance and stability of the algorithm for finding good seed sets.

5 Conclusion

In this work we focused on the Influence Maximization Problem. We develop a binary-integer program to represent
IMP with the Independent Cascade diffusion model. The original form of IMP is not favorable to be solved exactly, so we first converted it into a stochastic k-coverage problem and used the Sample Average Approximation (SAA) scheme to solve it. Then we proposed LP relaxation based pipage method with polynomial running time, which guarantees a $1 - 1/e - \epsilon$ worst-case optimality bound asymptotically. Experimental results over different networks and seed set values indicate that, the LP relaxation of IMP is very strong and the pipage method works very well and provides close-to-optimal solutions in most of the cases. Another important observation is the unexpectedly good performance of the Greedy and IMM methods, where both methods provide close-to-optimal solutions. In terms of running time performance, the pipage method is comparable with the greedy method and runs faster when the seed set size large. The superiority increases when the network size is larger and denser. As for the future research directions, the underlying LP structure can be analysed and its sparsity may be exploited. Faster methods to solve the LP relaxation such as column generation or Lagrangean relaxation might be promising as well.

REFERENCES


[15] Evren Gney received his B.S., M.S. and Ph.D. degrees in industrial engineering from Bogaziçi University, Istanbul, in 1999, 2002 and 2009 respectively. Currently he is an assistant professor in Department of Industrial Engineering in Istanbul MEF University. His research interests include operations research, integer programming, combinatorial optimization and social networks.


